**Aim Of the App**

The aim of the app is to identify the name of the song from an audio file of .wav extension and return information like song name, artist name, album and artist pic, and lyrics of the song. A database is created to store all the information about the song searched by the other users and is made accessible to all users in the form a Random song generator button which returns information about any random song from the database. The created web app should be responsive and adjust according to the screen size of the devices its being accessed from

**File Structure of the app**

The app has 4 major components:

1. UI/UX Component

* Templates Folder (contains all HTML)

Home.html

Results.html

Random.html

* Static Folder (contains all CSS)

Home.css

Results.css

1. The Processing component

Shazam\_core.py

Random\_song.py

1. Relational Database

Song\_data.db

1. App.py (which join both UI/UX with processing component)

**UI/UX of the app**

Home page

The home page consists of 2 major components.

1. Navbar

It is a Simple navbar containing 2 elements

* Navbar Title

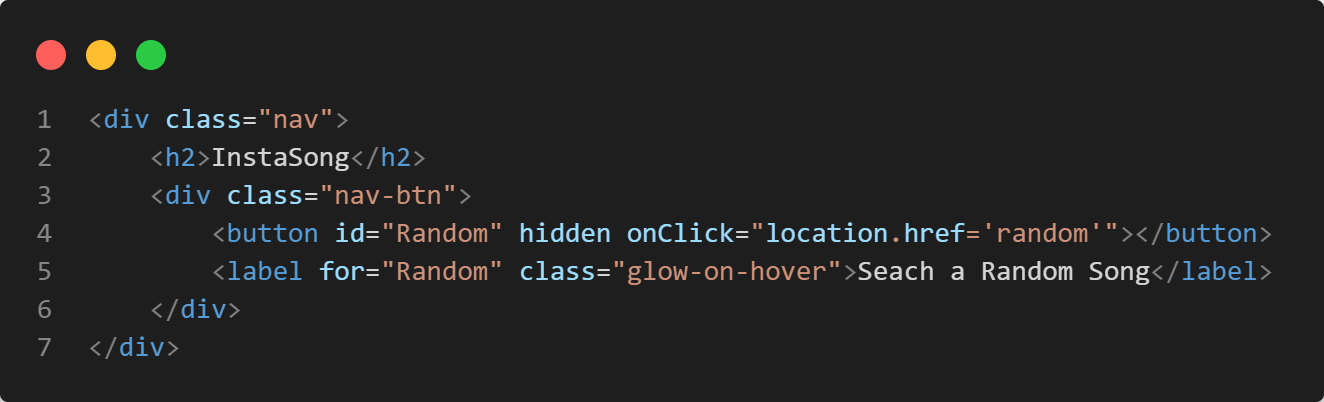
It is the name of the Project

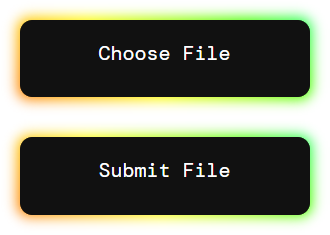
* Random Song Button

It output a random song which is stored in the database which was previously searched by the current or even another user.



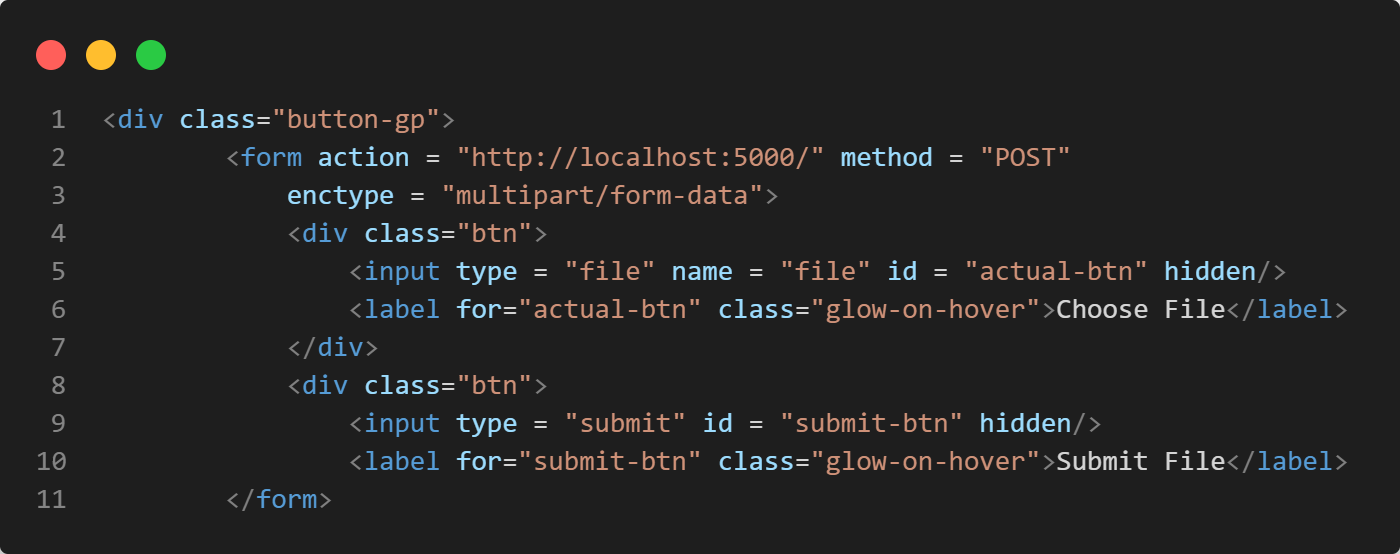
It uses the following code:



1. The Input and Submit Button Group

It takes the input from the user in the form of a .wav audio file of length 3-5 sec and passes the file in the form of POST request

It uses the following code:

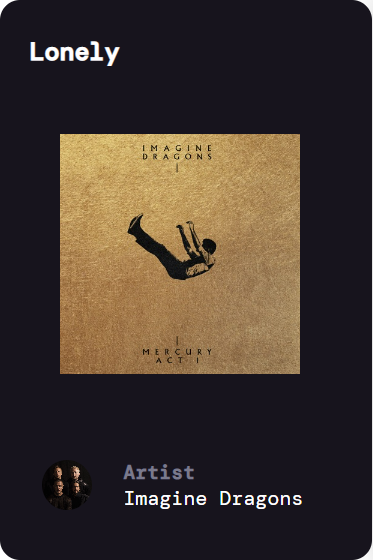


Results Page

The results page shows the track name, artist name, artist picture, album art and the lyrics for the song in the file uploaded by the user. It is divided in two components.

1. Song Information Card

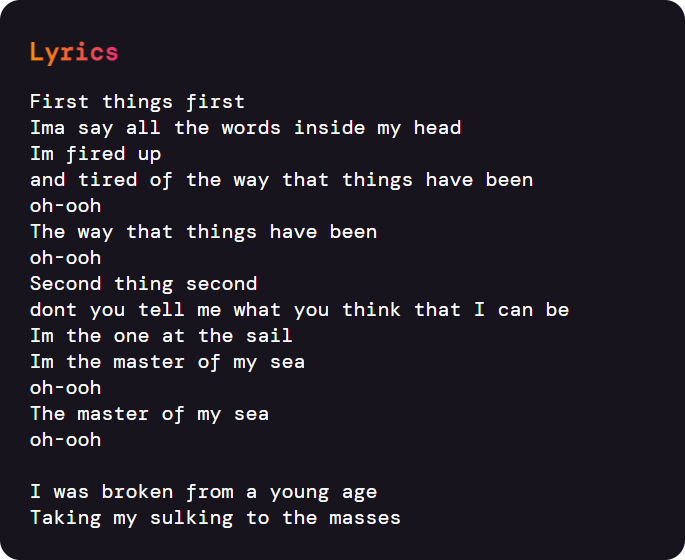
This element contains the track name, artist name, artist picture and the album art extracted from the output Shazam core API.



Song Information card uses the following html code



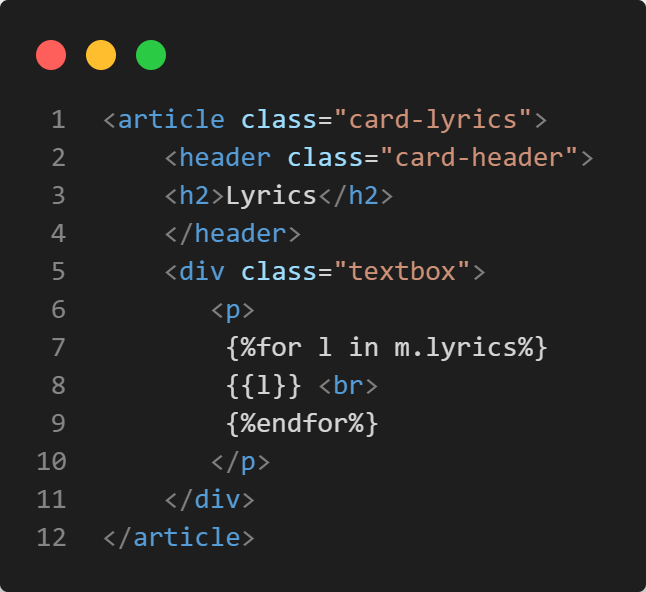
In the above code m is the dictionary which contains the information about the song.



2. Lyrics Card

This element contains the

Lyrics of the song extracted from the output Shazam core API.

Song Information card uses the following html code

In the above code m is the dictionary which contains the information about the song.

Random Page

The UI of this page is same as that of the results page. The key difference is that it gets the data from the connected SQlite3 database which contain the data of the songs previously searched by other users

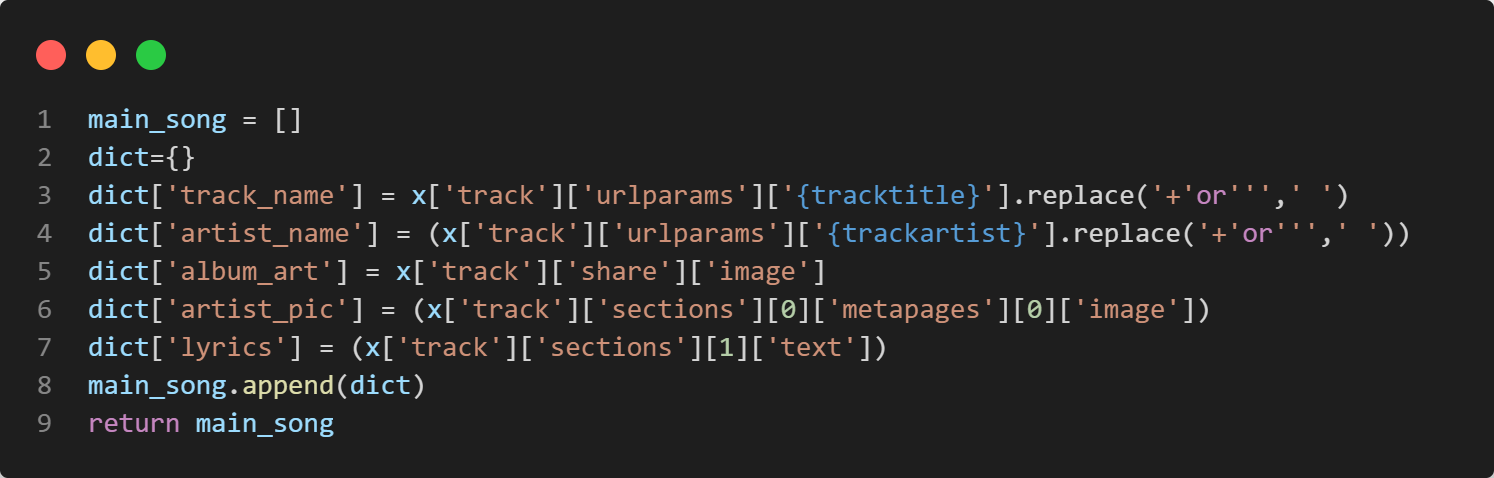
**The Processing Component**

Shazam\_core.py

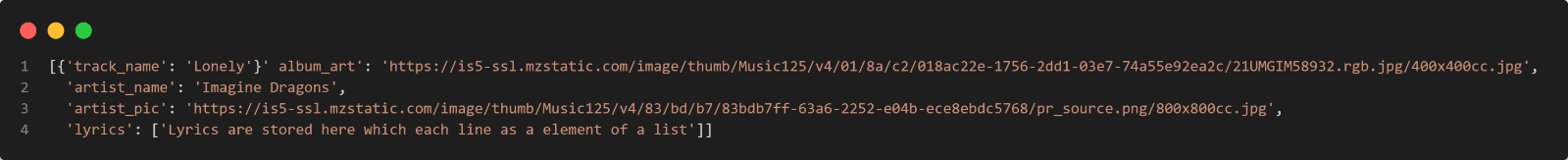
This file, when provided the input file from the user calls the Shazam core API by sending the .wav format file to the server and then getting a response in json format



The response from the server averages around 750 – 850 words. So, it needs to be shortened down. Useful data is extracted from the response like following

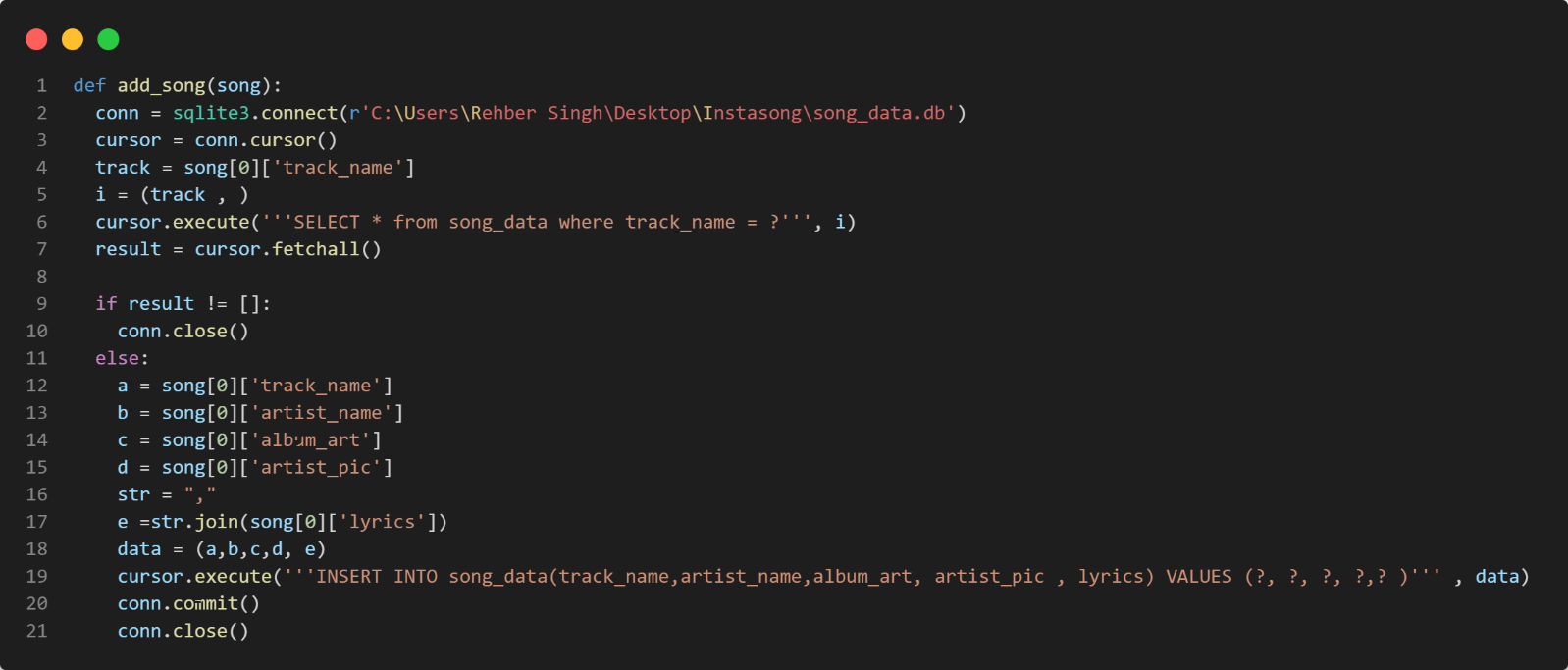


The output looks something like this

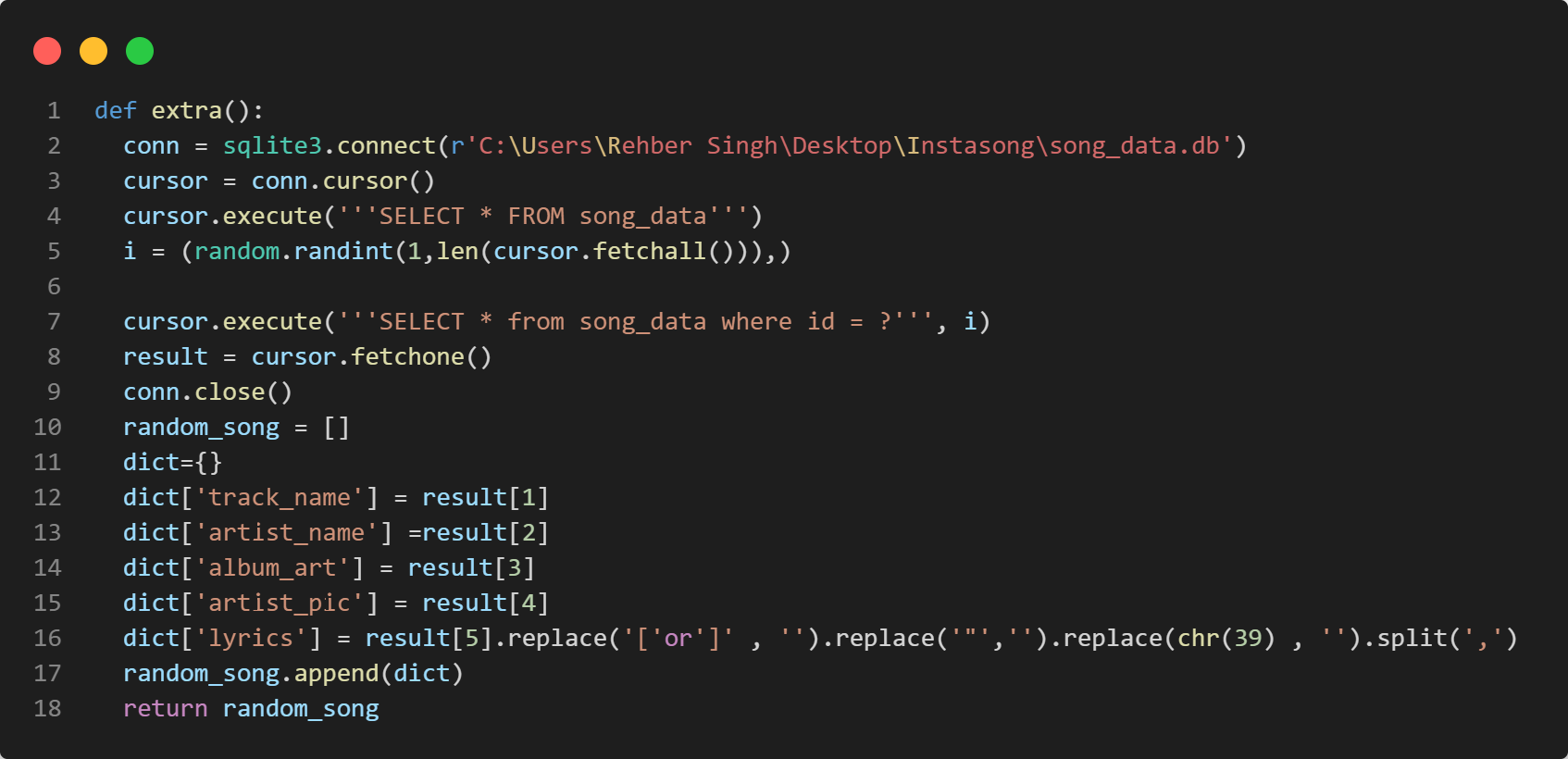
\*Lyrics are removed here to shorten the size of the image

Random\_Song.py

This file serves 2 purposes. Firstly, whenever a user searches for a song ever successful response is send to this file. If the response is unique and is not present in the database it is added to it by the following code.



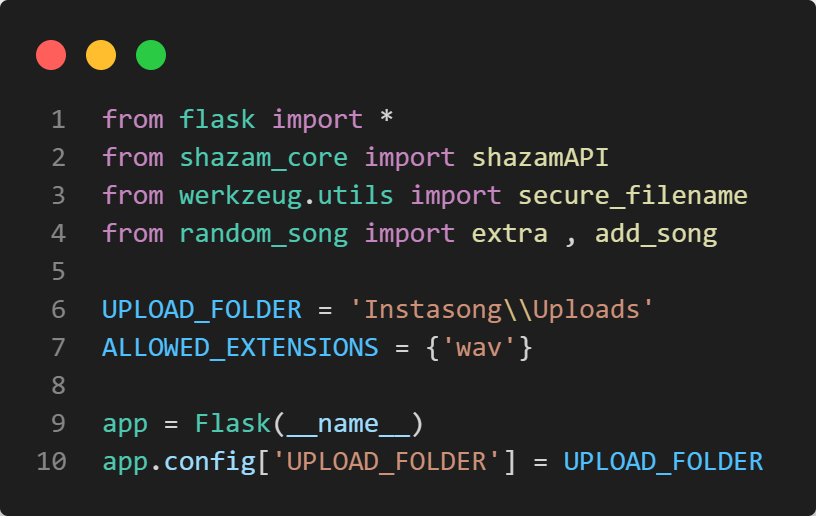
The second purpose of this file is to serve the data to the ramdom.html file whenever ‘Search a random Song’ button is pressed in the Navbar. This is achieved by the following piece of code

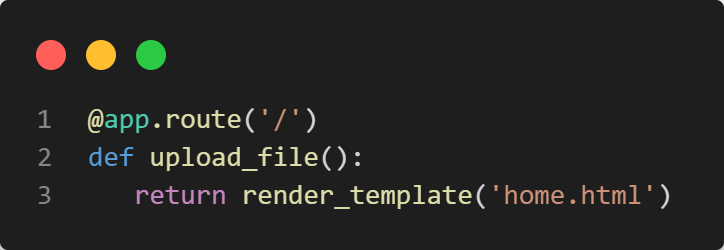


App.py

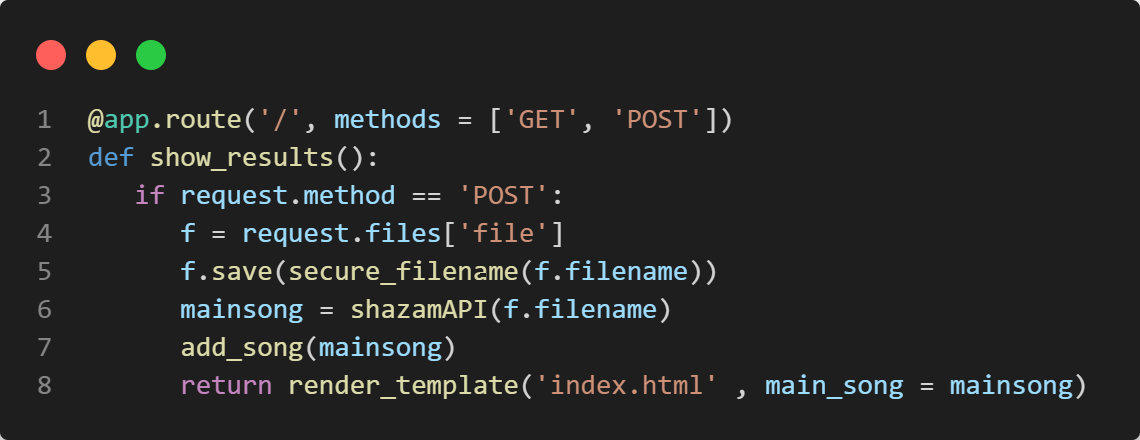
All the above function will make no sense if there is no way to connect the all. That is where this file comes in. It is the backbone of the application and all the main functions are called here.

The following code imports the necessary libraries and initializes the app



To create a basic structure of the web app routes are created. For the home page following piece of code render the home.html file.

On the home page the user can perform two actions either uploading a file of searching for random song. If user decides to upload a GET requests changes to post request and the uploaded file is passed to appropriate functions.



In the above code the uploaded file is passes onto the shazamAPI() function. This function returns a list in song info. This info is then passed onto add\_song() which adds it to song\_info database.

If the user decides to search for a random song a new webpage is open by following code



The above code calls extra() function which return song info in the form of a list from Song info database.